REMARKS

No claims have been amended. Claims 1-69 remain in the application for consideration. In view of the following amendments and remarks, Applicant respectfully solicits allowance of the application and furtherance onto issuance.

§ 102 Rejections

Claims 1-69 stand rejected under 35 U.S.C § 102(e), as being anticipated by U.S. Patent No. 6,253,366 to Mutschler, III (hereinafter "Mutschler").

Claims 1-16

Claim 1 recites a method comprising [emphasis added]:

- describing one or more software extensions using descriptions, the extensions being configured for incorporation in a software platform executing on a client; and
- delivering the descriptions of the one or more extensions to the client via a network, the descriptions being configured for use in downloading the software extensions via the network;
- said acts of describing and delivering being configured to enable ***software*** to be delivered over the network.

In making out the rejection of this claim, the Office cites to column 2, lines 19-22 and 27-31, of Mutschler, reproduced below [emphasis added]:

> Another object of the present invention is to provide a method and system that allows developers of distributed systems the ability to share ***object models*** and ***other metadata*** over a network, including the Internet. *Col. 2, lines 19-22.*
>
> A feature of the present invention is the use of entity objects to encapsulate properties and behaviors of each class object thereby making the document type definition (DTD) more compact and giving a clearer picture of the ***relationships*** in the meta-model being captured. *Col. 2, lines 27-31.*

The Office argues that "Google.com defines an object model: 'An object model is a collection of descriptions of classes or interfaces, together with their member data, member functions and class-static operations.' Thus, Examiner maintains that the Mutschler reference does apply to delivering software over a network."

Applicant agrees that the definition cited above can be found on Google. However, the Office's cited definition states that an object model is a collection of ***descriptions***. For a more complete understanding of what an object model is, Applicant respectfully directs the Office's attention to the entire page of definitions found through Google [emphasis added]:

> Definitions of **object model** on the Web:
>
> An object model is a collection of ***descriptions*** of classes or interfaces, together with their member data, member functions, and class-static operations.
> www.w3.org/TR/1998/WD-DOM-19980720/glossary.html
>
> A computer representation that encapsulates data attributes and behavioral processes (operations) for an object. Object model software may respond to events, triggers, and requests for service submitted as message stimuli (with a finite set of message types, argument types and message formats). An object model is a ***graphical representation of the structure of objects*** in a system including their: identity, attributes, operations, ***and associations between objects***.
> info.louisiana.edu/dept/gloso.html
>
> In object-oriented programming languages, the design of an object and the classes required to create and enable an instance of the object by using methods, properties, and events to interact with the object.
> www.microsoft.com/technet/prodtechnol/project/project2000/reskit/projglos.asp
>
> A ***description*** of the ***structural relationships*** among components of a library object including its metadata.
> www.cs.cornell.edu/wya/DigLib/MS1999/glossary.html
>
> The model that reflects as objects the overall object-oriented ***design*** of an application or system.
> edocs.bea.com/wle/wle42/glossary/glossary.htm
>
> "Current leading object models are primarily focusing on document usage. They simplify the creation and management of compound documents, where elements of the document are created and maintained by diverse applications. [ ]

Distributed object models enable the objects to be used across the network, and, additionally, have facilities for object activation and information passing." www.for.gov.bc.ca/isb/datadmin/glos_n.htm

The ***conceptual representation*** of the problem domain of an application that embodies the business rules being automated. An object model, ***typically represented with a class diagram***, is used to validate requirements, drive a software solution (object design) or to re- engineer the business rules. www.carolla.com/glossary.html

A data model derived from object-oriented programming that encapsulates data and methods and organizes objects into object classes, among which there can be a hierarchical relationship. lms.thomsonelearning.com/hbcp/glossary/glossary.taf

In OMT terminology (that we use), Object Model refers to the ***structure of objects*** in a system. It is ***graphically represented using class diagrams*** depicting the classes that the objects in a system belong to and the relationships among them. www.magnetar.org/glossary.htm

The object oriented design model has several aspects: abstraction, encapsulation, modularity, hierarchy, typing, concurrency, and persistence. www.cs.uwa.edu.au/programming/c++.tutorial/glossary/

A specification of the objects intrinsic to a given system, including a ***description of the object characteristics*** (attributes) and a ***description*** of the static and dynamic ***relationships*** that exist between objects. www.semb.co.uk/reference/glossary.htm

a model in terms of objects and their associated relationships. Contrast with business model and use case model. www.donald-firesmith.com/Glossary/GlossaryO.html

A ***conceptual map*** for the hierarchical chain of objects that are exposed by an application. highered.mcgraw-hill.com/sites/0072470925/student_view0/glossary.html

A collection of objects having properties and methods that provide a specialization namespace for describing a system and its functionality. In the case of the Channel Server, the namespace is based on a channel metaphor. (Push) www.utpb.edu/siteserver/docs/cc_gloss_abca.htm

The structural foundation of an object-oriented language, design, or application. Comprises an object architecture's ***description***, including details of the object structure and interfaces between objects. www.cobycoinc.com/WebTools/cdfiles/morecdfiles/articles/glossary.htm

In addition, there are a multitude of resources available, both in print and online, that discuss object models.

Applicant strongly disagrees with the Office equating Mutschler's object model with Applicant's ***software***. Even the definition that the *Office itself* cites to

defines an object model as a collection of ***descriptions*** of classes or interfaces. Furthermore, numerous other definitions available through Google, and elsewhere, specify that an object model is a ***graphical representation*** of the structure of objects and their relationship to one another. This is quite different from ***software***. Applicant has thoroughly reviewed the reference and respectfully submits that ***nowhere*** does Mutschler teach ***software*** delivery over a network. Accordingly, for at least this reason, this claim is allowable.

Claims 2-16 depend from claim 1 and are allowable as depending from an allowable base claim. These claims are also allowable for their own recited features which, in combination with those recited in claim 1, are neither disclosed nor suggested in the references of record, either singly or in combination with one another.

Claim 17

Claim 17 recites one or more computer-readable media having computer-readable instructions thereon which, when executed by a computer system, cause the computer system to [emphasis added]:

- describe one or more software extensions using extensible markup language (XML), the extensions being configured for incorporation in a software platform comprising a single application program, the single application program having multiple different functionalities that can enable a user to accomplish multiple different tasks; and
- deliver XML descriptions of the one or more extensions to the client via the Internet, the descriptions being configured for use in downloading the software extensions via the Internet;
- wherein causing said computer system to describe one or more extensions and deliver XML descriptions enables ***software*** to be delivered over the Internet.

In making out the rejection of this claim, the Office again cites to column 2, lines 21-22 and 27-31, reproduced above, to support its argument that Mutschler teaches delivery of software over the Internet.

Applicant strongly disagrees with the Office equating Mutschler's object model with Applicant's ***software***. Even the definition that the *Office itself* cites to defines an object model as a collection of ***descriptions*** of classes or interfaces. Furthermore, numerous other definitions available through Google, and elsewhere, specify that an object model is a ***graphical representation*** of the structure of objects and their relationship to one another. This is quite different from ***software***. Applicant has thoroughly reviewed the reference and respectfully submits that ***nowhere*** does Mutschler teach ***software*** delivery over a network. Accordingly, for at least this reason, this claim is allowable.

Claims 18-28

Claim 18 recites a method comprising [emphasis added]:

- describing one or more software extensions using one or more descriptive files, the extensions being configured for incorporation in a software program executing on a client;
- associating the one or more descriptive files with one or more associated extension files that are useable to provide a ***program functionality***;
- storing the descriptive files and associated extension files in a network-accessible location; and
- delivering the descriptive files and the associated extension files of the one or more extensions to the client via a network.

In making out the rejection of this claim, the Office again cites to column 2, lines 19-22, of Mutschler, reproduced above. The Office appears to argue that the

description of an object's operation in a model provides program functionality. Applicant disagrees. As discussed above, an object model is a ***graphical representation*** of the structure of objects and their relationship to one another. The mere fact that the operation is ***described*** in a model as being associated with a particular object does not provide program functionality. It provides ***information*** about the operation, but the model does not ***invoke*** the operation through mere transfer of the model itself. Because Mutschler's model does not invoke the operation, the described operation cannot possibly provide program functionality. Accordingly, for at least this reason, this claim is allowable.

Claims 19-28 depend from claim 18 and are allowable as depending from an allowable base claim. These claims are also allowable for their own recited features which, in combination with those recited in claim 18, are neither disclosed nor suggested in the references of record, either singly or in combination with one another.

Claims 29-39

Claim 29 recites a method comprising [emphasis added]:

- storing one or more extension definition files (EDFs) that describe a logical attachment to a software application program;
- storing one or more extension files that correspond to the one or more EDFs and extend the software application program;
- delivering, via a network, at least one EDF to a client; and
- delivering, via the network, at least one extension file that corresponds to the at least one EDF to a client;
- both of said acts of storing and both of said acts of delivering enabling ***software*** to be delivered over the network.

In making out the rejection of this claim, the Office again cites to column 2, lines 20-22, of Mutschler, reproduced above.

Applicant strongly disagrees with the Office equating Mutschler's object model with Applicant's ***software***. Even the definition that the *Office itself* cites to defines an object model as a collection of ***descriptions*** of classes or interfaces. Furthermore, numerous other definitions available through Google, and elsewhere, specify that an object model is a ***graphical representation*** of the structure of objects and their relationship to one another. This is quite different from ***software***. Applicant has thoroughly reviewed the reference and respectfully submits that ***nowhere*** does Mutschler teach ***software*** delivery over a network. Accordingly, for at least this reason, this claim is allowable.

Claims 30-39 depend from claim 29 and are allowable as depending from an allowable base claim. These claims are also allowable for their own recited features which, in combination with those recited in claim 29, are neither disclosed nor suggested in the references of record, either singly or in combination with one another.

Claims 40-47

Claim 40 recites a data structure embodied on a computer-readable medium comprising [emphasis added}:

- a first sub-structure indicative of a software extension that is to be incorporatcd in a software application program;
- one or more second sub-structures associated with the first sub-structure and indicating ***feature types*** that are added by the extension to the application program; and

- one or more third sub-structures associated with the one or more second sub-structures and indicating features of an associated ***feature type*** that are added by the extension.

In making out the rejection of this claim, the Office cites to column 6, lines 35-36, 41-45, and 50-57, and column 14, lines 7-8, 33, and 42-52. These excerpts are set forth below [emphasis added]:

> There are various methods by which the DTD generator 19 can produce the DTD (bubble 19A). *Col. 6, lines 35-36.*
>
> As the DTD productions in the first above-cited co-pending patent application (hereafter referred to as the "First Rule Set") are very simple, they can result in large DTD's. The repetition of detail also makes it difficult to perform modifications for the purposes of extension or experimentation. *Col. 6, lines 41-45.*
>
> The method of the present invention allows for grouping of the parts of an object into XML entity definitions. These entities may be used in place of the actual listing of the elements. This makes for more compact DTD files. The savings is about 15 to 20 percent over that of the First Rule Set. In addition, since the Attributes, References and Compositions of an object are defined in only one place, modification is greatly simplified. *Col. 6, lines 50-57.*
>
> Referring now to FIG. 16A, the first of a two-sheet flow chart of the Compositions Entities Definitions 31 process is illustrated. *Col. 14, lines 7-8.*
>
> Referring now to FIG. 16B at the connector J, an inquiry is made as to whether or not there are more sub-Classes for the Class (diamond 231). *Col. 14, line 33.*
>
> Auxiliary functions are required for several purposes, among which are the recursive procedures to manage inheritance and for XML details. The code for implementing the auxiliary functions is set forth in Exhibit A hereof.
>
> These functions illustrate possible methods to ***perform the textual manipulations necessary to insure that the formatting of the XML***

> ***definitions is correct***. They also illustrate possible methods to obtain lists of Attributes, Classes, etc., where Class or Package inheritance is involved. While these functions can be used to perform the indicated operation, they are not necessarily the only means of so doing. *Col. 14, lines 42-52.*

Specifically, the Office makes reference to column 14, lines 42-52, and appears to argue that Mutschler's formatting of XML definitions is somehow equivalent to Applicant's feature types, namely "menu items, style sheets, etc."

Applicant strongly disagrees. Mutschler's auxiliary functions are designed to perform ***textual manipulations*** of ***XML definitions*** so that the XML definitions are ***formatted*** correctly. This is quite different from Applicant's feature types. The Office's attention is respectfully drawn to Applicant's specification, page 14, lines 12-19, in addition to Table 1, for a discussion of feature types. This excerpt is reproduced below for the Office's convenience [emphasis added]:

> EDFs advantagcously have an "open schema" which means that third party developers can extend the extension mechanism and include their own extensions by creating their own tags. Additionally, extensions can themselves be extended by other developers. EDFs can also have one or more predefined tags. Exemplary predefined XML tags for user interface elements can include tags for ***feature types such as: tool bars, accelerators, menu items, and themes***. These feature types are utilized in the single navigable window application incorporated by reference above and defined in the table immediately below:

Feature Type	Definition
Tool Bars	Horizontal command containers above the document area.
Accelerators	Keyboard shortcuts for commands
Menu Items	Pop-up or drop-down menu choices that third parties can add to well-known, named menu attachments in the platform
Themes	A data-driven way to provide overrides for well-known resources of the platform, such as default buttons or default style sheet

Table 1

Applicant respectfully submits that there is no relationship *whatsoever* between Mutschler's formatting of XML definitions and Applicant's feature types. Accordingly, for at least this reason, this claim is allowable. It is to be appreciated that the feature types illustrated in the excerpt above are but mere *examples* of what is meant by a "feature type" within the context of this claim. If the Office insists upon maintaining this rejection, Applicant respectfully requests the Office to *specifically* point out where Mutschler teaches "feature types" as Applicant has defined and used the term.

Claims 41-47 depend from claim 40 and are allowable as depending from an allowable base claim. These claims are also allowable for their own recited features which, in combination with those recited in claim 40, are neither disclosed nor suggested in the references of record, either singly or in combination with one another.

Claims 48-53

Claim 48 recites a method of delivering software via a network comprising [emphasis added]:

- navigating to a network site that maintains at least one software application program; and
- downloading a ***software application program*** from the network site, the application program comprising multiple different functionalities that can assist a user in accomplishing different tasks, the ***software application program*** being configured to be extended with software extensions that are deliverable via a network and are described by at least one network-deliverable file.

In making out the rejection of this claim, the Office cites to column 1, lines 49-52, and column 2, lines 19-22 and 27-31, of Mutschler, reproduced above. In addition, the Office again cites to the same Google definition of object model.

Applicant strongly disagrees with the Office equating Mutschler's object model with Applicant's ***software*** or with Applicant's ***software application program***. Even the definition that the *Office itself* cites to defines an object model as a collection of ***descriptions*** of classes or interfaces. Furthermore, numerous other definitions available through Google, and elsewhere, specify that an object model is a ***graphical representation*** of the structure of objects and their relationship to one another. This is quite different from downloading a ***software application program***. Applicant has thoroughly reviewed the reference and respectfully submits that ***nowhere*** does Mutschler teach downloading a ***software application program*** with multiple different functionalities that can assist a user in accomplishing different tasks. Nor does Mutschler teach that a ***software application program*** can be extended with software extensions that are deliverable via a network. Accordingly, for at least these reasons, this claim is allowable.

Claims 49-53 depend from claim 48 and are allowable as depending from an allowable base claim. These claims are also allowable for their own recited

features which, in combination with those recited in claim 48, are neither disclosed nor suggested in the references of record, either singly or in combination with one another.

Claim 54

Claim 54 recites one or more computer-readable media having computer-readable instructions thereon which, when executed by a computer, cause the computer to [emphasis added]:

- navigate to a network site that maintains at least one software application program;
- download a ***software application program*** comprising multiple different functionalities that can assist a user in accomplishing different tasks, the software application program being configured to be extended with software extensions that are deliverable via the network and described by at least one network-deliverable file; and
- extend the software application program by adding at least one extension to the application program, the extension being added by using a link to navigate to a different network site that hosts one or more files that describe the extension, and extension files that are used to implement the extension and downloading the one or more files and the extension files to a client.

In making out the rejection of this claim, the Office cites to the same sections of Mutschler as were cited to in making out the rejection of claim 48. In addition, the Office again cites to the same Google definition of object model.

Applicant strongly disagrees with the Office equating Mutschler's object model with Applicant's ***software*** or with Applicant's ***software application program***. Even the definition that the *Office itself* cites to defines an object model as a collection of ***descriptions*** of classes or interfaces. Furthermore, numerous

other definitions available through Google, and elsewhere, specify that an object model is a ***graphical representation*** of the structure of objects and their relationship to one another. This is quite different from downloading a ***software application program***. Applicant has thoroughly reviewed the reference and respectfully submits that ***nowhere*** does Mutschler teach downloading a ***software application program*** with multiple different functionalities that can assist a user in accomplishing different tasks. Nor does Mutschler teach that a ***software application program*** is configured so that it can be extended with software extensions that are deliverable via the network. Accordingly, for at least these reasons, this claim is allowable.

Claims 55-62

Claim 55 recites a method comprising [emphasis added]:

- accessing a Web site through which one or more software extensions can be obtained and through use of which ***software*** can be delivered;
- receiving at least one file that describes at least one software extension using a hierarchical language that describes the software extension's logical attachment to a software application program;
- receiving one or more software extension files; and
- installing the one or more software extension files based, at least in part, on the description contained in said at least one file.

The Office contends that the subject matter of this claim is disclosed in column 4, lines 21-39, and column 6, lines 11-16 and 29-49 of Mutschler. However, as discussed above, Mutschler does not teach, in these excerpts or anywhere else, software extensions that can be obtained and through use of which

software can be delivered. Accordingly, for at least this reason, this claim is allowable.

Claims 56-62 depend from claim 55 and are allowable as depending from an allowable base claim. These claims are also allowable for their own recited features which, in combination with those recited in claim 55, are neither disclosed nor suggested in the references of record, either singly or in combination with one another.

Claim 63

Claim 63 recites a method comprising [emphasis added]:

- describing one or more software extensions using one or more extensible markup language (XML) files, the extensions being configured for incorporation in a software program executing on a client;
- associating the one or more XML files with one or more associated extension files that are useable to provide a program functionality; and
- storing the XML files and associated extension files in a network-accessible location;
- said acts of describing and associating being configured to provide *software* for delivery over the network.

In making out the rejection of this claim, the Office again cites to column 2, lines 20-22, of Mutschler, reproduced above.

Applicant strongly disagrees with the Office equating Mutschler's object model with Applicant's *software*. Even the definition that the *Office itself* cites to defines an object model as a collection of ***descriptions*** of classes or interfaces. Furthermore, numerous other definitions available through Google, and elsewhere, specify that an object model is a ***graphical representation*** of the structure of

objects and their relationship to one another. This is quite different from ***software.*** Applicant has thoroughly reviewed the reference and respectfully submits that ***nowhere*** does Mutschler teach ***software*** delivery over a network. Accordingly, for at least this reason, this claim is allowable.

Claims 64-65

Claim 64 recites a network site comprising:

- one or more software extension files configured to be incorporated into a software application program, the software extension files being configured to allow delivery of ***software*** via a network; and
- one or more files associated with the one or more software extension files and describing the extension files, the one or more files describing a logical attachment of the one or more software extension files to the software application program.

The Office contends that the subject matter of this claim is disclosed in column 4, lines 21-39, and column 6, lines 11-16 and 29-49. However, as discussed above, Mutschler does not teach, in these excerpts or anywhere else, software extensions being configured to allow delivery of ***software*** via a network. For at least this reason, this claim is allowable.

Claim 65 depends from claim 64 and is allowable as depending from an allowable base claim. This claim is also allowable for its own recited features which, in combination with those recited in claim 64, are neither disclosed nor suggested in the references of record, either singly or in combination with one another.

Claims 66-69

Claim 66 recites a method of managing network-based software extensions comprising [emphasis added]:

- grouping multiple software extension descriptions in a catalog in a network-accessible location to enable delivery of ***software*** via a network;
- accessing the network-accessible location; and
- using the catalog to update a software extension that is resident on a computing device.

In making out the rejection of this claim, the Office cites to column 5, lines 16-23, and column 6, lines 11-12 and 22-36. However, as discussed above, Mutschler does not teach, in these excerpts or anywhere else, grouping multiple software extension descriptions in a catalog in a network-accessible location to enable delivery of ***software*** via a network. For at least this reason, this claim is allowable.

Claims 67-69 depend from claim 66 and are allowable as depending from an allowable base claim. These claims are also allowable for their own recited features which, in combination with those recited in claim 66, are neither disclosed nor suggested in the references of record, either singly or in combination with one another.

Conclusion

Applicant has studied the reference cited by the Office and has sincerely attempted to describe how the claimed subject matter patentably distinguishes over this reference. Applicant submits that all of the claims are in condition for

allowance and respectfully requests that the Office pass the application along to issuance. If the Office's next anticipated action is to be anything other than issuance of a Notice of Allowability, Applicant respectfully requests a telephone call for the purpose of scheduling an interview.

Respectfully Submitted,

Dated: 8/16/04 By: RR Cottle

Rob R. Cottle
Reg. No. 52,772
(509) 324-9256

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Application Serial No. 09/599,048
Filing Date 6/21/2000
Inventorship ..... Murray et al.
Applicant ..... Microsoft Corp.
Group Art Unit 2122
Examiner ..... Steelman, Mary J
Attorney's Docket No. MS1-563US
Title: Network-based Software Extensions

INTERVIEW SUMMARY FOR INTERVIEW CONDUCTED JULY 8, 2004 REQUIRED UNDER 37 CFR 1.133(b)

To: Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

From: Rob R. Cottle (Tel. 509-324-9256, ext 247; Fax 509-323-8979)
Lee & Hayes, PLLC
421 W. Riverside Avenue, Suite 500
Spokane, WA 99201

ADMINISTRATIVE DETAILS

Status: Not Under Final

Participants: Rob Cottle and Mary Steelman

Proposed Date of Interview: July 8, 2004

Actual Date of Interview: July 8, 2004

Type of Interview: Telephonic

Exhibit Shown: No

ISSUES DISCUSSED

Applicant and the examiner discussed the Mutschler reference (U.S. Patent 6,253,366), which is used to reject **claims 1-56** in the outstanding Office Action.

Arguments Made and Agreements Reached

The outstanding Office Action cites to a Google definition of the term "object model" to support the examiner's argument that Mutschler teaches software delivery. Applicant argued that Mutschler's object model is not the same as Applicant's ***software*** delivery. Applicant explained that an object model is a diagrammatic representation of the structure and relationship between objects.

The outstanding Office Action cites column 14, lines 42-53, to support its argument that Mutschler teaches feature types. Applicant argued that Mutschler does not teach feature types, as Applicant defines and uses the term. Applicant explained that the cited portion deals merely with functions for manipulating text so that Mutschler's XML definitions are formatted correctly. Applicant referred to the examiner to Applicant's specification for a definition of the term "feature types."

The examiner agreed to read the Mutschler reference more carefully and to get input from a colleague regarding these issues.

Respectfully Submitted,

Dated: 7/26/04

By: RR Cottle

Rob R Cottle
Reg. No. 52,772
(509) 324-9256 ext. 247